

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY  
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY



**SOICT**

# PROJECT REPORT

## MICROSOFT TEAM MANAGEMENT APP

**Course:** Project I

**Supervisor:** Tong Van Van

Members	Student ID
Truong Thao Nguyen	20220067
Phan Thu Ha	20225571

July, 2024

# Contents

1	Introduction . . . . .	3
2	Methodology . . . . .	4
2.1	APIs . . . . .	4
2.1.1	Microsoft Graph, Access Token, Azure App Permissions and Secrets . . . . .	4
2.1.2	Airtable API . . . . .	5
2.2	Java Programming . . . . .	5
2.2.1	Usecase Diagram . . . . .	5
2.2.2	Class Diagram . . . . .	5
2.2.3	Deploying with Jenkins and testing with SonarQube . . . . .	6
3	Product Demo . . . . .	10
3.1	Verify permission . . . . .	10
3.2	Sync data to your Airtable base . . . . .	11
3.3	Create a channel . . . . .	12
3.4	Add new members to channel . . . . .	12
3.5	Add new members to Team from CSV files . . . . .	13
4	Future Development Discussion . . . . .	14
4.1	More user-friendly GUI . . . . .	14
4.2	Smarter Syncing . . . . .	14
5	Conclusion . . . . .	15

# 1 Introduction

In today's fast-paced business environment, effective team management and seamless communication are critical for success. Microsoft Teams has emerged as a leading platform for collaborative work [1], offering a wide array of features to facilitate communication, collaboration, and organization. However, managing a Microsoft Teams environment can be complex, particularly for large organizations with numerous teams and channels.

To address these challenges, we have developed a comprehensive Microsoft Team Management app designed to streamline and automate key management tasks. Our app focuses on solving several critical problems to enhance efficiency and ensure proper governance within Microsoft Teams:

- **Ensuring Proper Permissions:** Our app ensures that specific permissions are granted to the appropriate employees, enabling precise and secure management of team settings and resources.
- **Data Synchronization with Airtable:** Keeping track of all data related to team members and channels can be daunting. Our app synchronizes this data with an Airtable base, providing a unified and accessible view of all team-related information.
- **Bulk Member Addition from CSV Files:** Adding new members to a team can be time-consuming, especially when dealing with large numbers. Our app simplifies this process by allowing bulk addition of members from CSV files, saving time and reducing errors.
- **Adding Members to Private Channels:** Managing private channels and their memberships can be challenging. Our app streamlines this process, making it easy to add specific members to private channels as needed.
- **Channel Creation:** Whether creating standard, private, or shared channels, our app provides a straightforward interface to create new channels efficiently, ensuring they meet the specific requirements of the team.

By addressing these critical aspects of Microsoft Teams management, our app enhances the overall user experience, improves organizational efficiency, and ensures that team management is both secure and streamlined. The following sections will delve deeper into the features and functionalities of our Microsoft Team Management app, illustrating how it can transform the way organizations manage their Microsoft Teams environments.

## 2 Methodology

### 2.1 APIs

#### 2.1.1 Microsoft Graph, Access Token, Azure App Permissions and Secrets

Our management application integrates with Microsoft Teams through the *Microsoft Graph API* [?], a comprehensive and unified endpoint for accessing data, intelligence, and insights from Microsoft 365 services. The Graph API provides programmatic access to various Microsoft services, allowing our app to manage teams, channels, and users efficiently.

To interact with the Graph API, our application must obtain an *access token* from the Microsoft identity platform. This access token is a security credential that verifies the app's identity and ensures it has the necessary authorization to access Microsoft Graph. The process involves authenticating the app and requesting an access token, which includes details about the app's permissions and scope of access. To ensure our app is properly authorized, we create an Azure application with the following *permissions*

- **Channel.Create** [3]: Allows the app to create new channels within a team.
- **ChannelMember.ReadWrite.All** [4]: Enables the app to read and update the members of any channel.
- **TeamMember.ReadWrite.All** [5]: Grants the app the ability to read and update team members.
- **TeamMember.Read.All** [6]: Allows the app to read team members' information.

These permissions are crucial as they enable our app to perform essential management tasks such as creating channels, adding or removing members, and retrieving necessary information about teams and channels.

As a security measure, our application requires team owners to provide the *Azure app's secret*. The app's secret is a credential used to authenticate the Azure application when requesting an access token from the Microsoft identity platform. By obtaining the Azure app's secret, our application can securely acquire the access token needed to interact with the Microsoft Graph API. [7]

This ensures that only authorized users can access and manage Microsoft Teams data through our app, maintaining a high level of security and control. By leveraging Microsoft Graph and adhering to robust authentication and authorization protocols, our app provides a seamless and secure solution for managing Microsoft Teams environments.

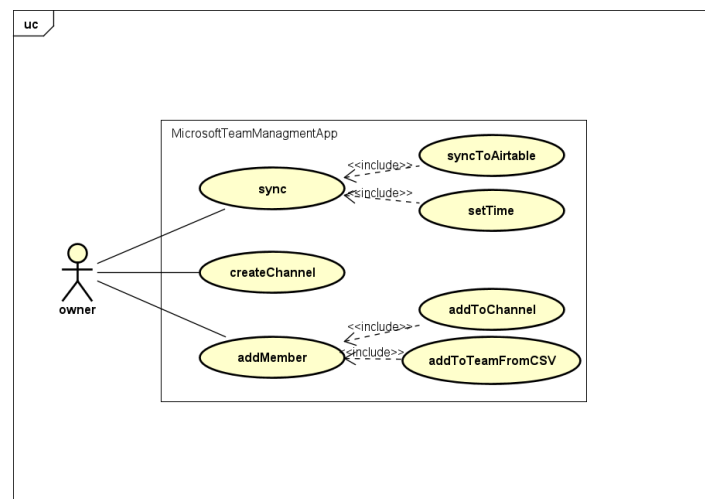
### 2.1.2 Airtable API

Airtable is a versatile cloud-based collaboration platform that combines the features of a spreadsheet and a database. It allows users to organize and manage data in a highly customizable and visually appealing manner. [8]

In the context of our project, we utilize Airtable to store information about members and channels retrieved from Microsoft Teams. This is facilitated by the *Airtable API*, a powerful and flexible tool that enables us to interact programmatically with Airtable bases. By leveraging the Airtable API, our app ensures seamless synchronization of data between Microsoft Teams and Airtable, enhancing data accessibility and management.

## 2.2 Java Programming

### 2.2.1 Usecase Diagram



### 2.2.2 Class Diagram

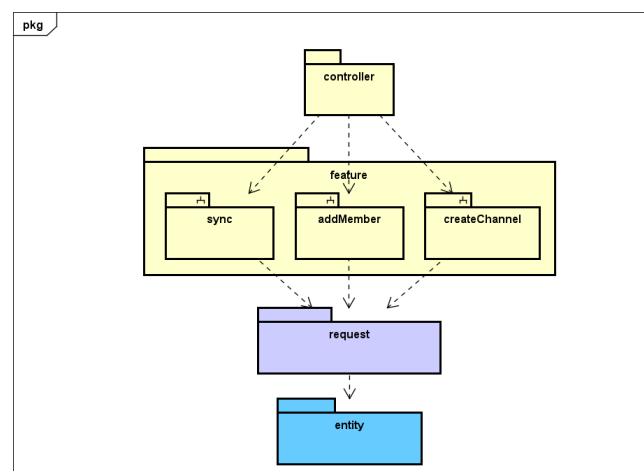


Figure 1: Package Dependency

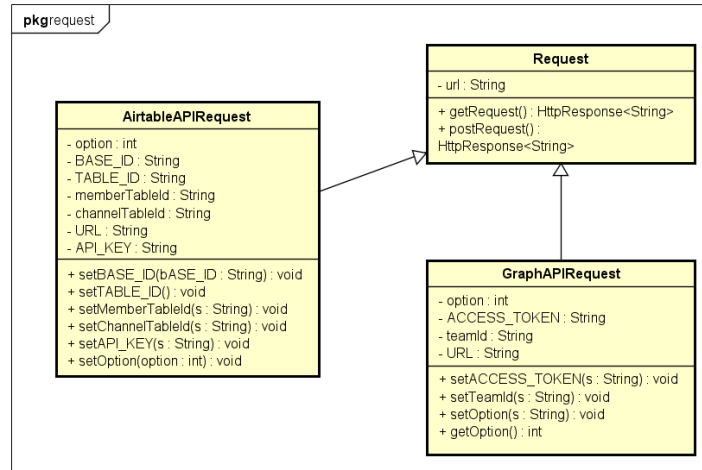


Figure 2: Classes in request package

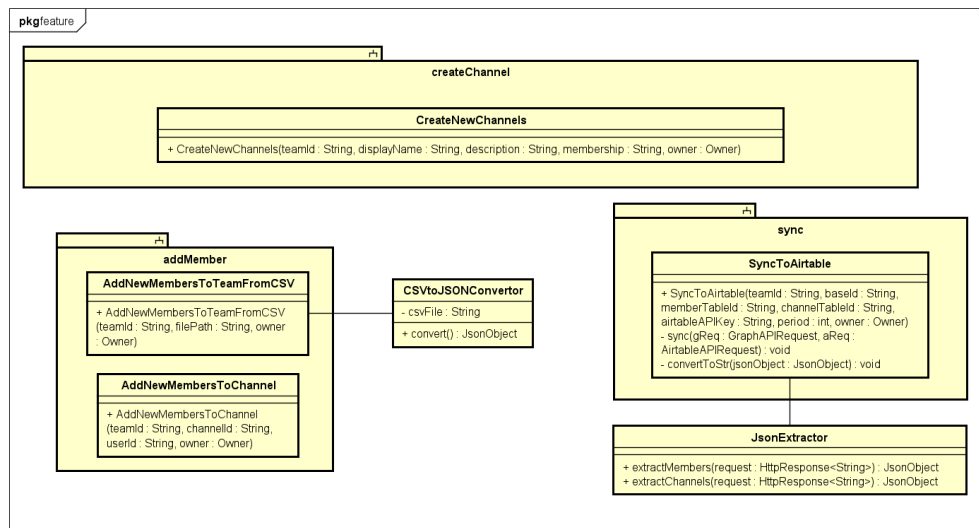


Figure 3: Subpackages and classes in Feature package

### 2.2.3 Deploying with Jenkins and testing with SonarQube

Jenkins [9] is an open-source automation server that facilitates continuous integration and continuous delivery (CI/CD). It allows developers to automate the building, testing, and deploying of applications. By integrating with various tools and platforms through a rich ecosystem of plugins, Jenkins helps streamline the software development lifecycle, ensuring that changes to the codebase are automatically tested and deployed. Jenkins pipelines, defined using the Jenkinsfile, describe the stages of the CI/CD process, enabling teams to maintain consistency and reproducibility in their workflows.

SonarQube [10] is an open-source platform designed for continuous inspection of code quality. It performs automatic reviews of code to detect bugs, vulnerabilities, and code smells in more than 25 programming languages. It also provides reports on duplications, coding standards, unit tests, code coverage, and code complexity.

Integrating Jenkins with SonarQube involves setting up Jenkins to trigger SonarQube analysis during the CI/CD pipeline. This integration allows automatic code quality

checks as part of the build process. A typical Jenkins pipeline would include stages for building the project, running SonarQube analysis, and executing tests. The results from SonarQube are then used to ensure that only code meeting the required quality criteria is deployed, thus maintaining high standards of code quality throughout the development process.

There are several bugs and code smells we have met in our project:

- **Abstract class without direct or inherited field:**

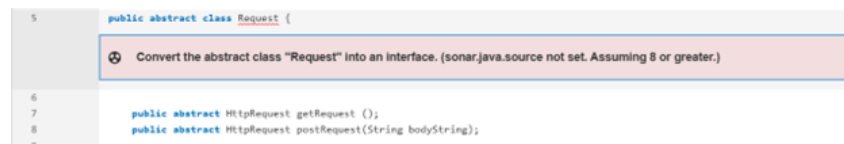


Figure 4: Abstract class without direct or inherited field should be converted into an interface, given Java 8 and over.

- **Casting in multiplication operation:**

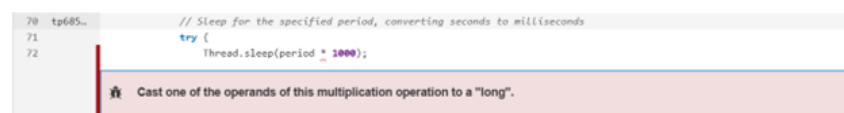


Figure 5: A least one operand should be cast or promoted to the final type before the operation takes place.

- **Commented-out lines:**



Figure 6: Programmers should not comment out code as it bloats programs and reduces readability. Unused code should be deleted.

- Debug feature activated:

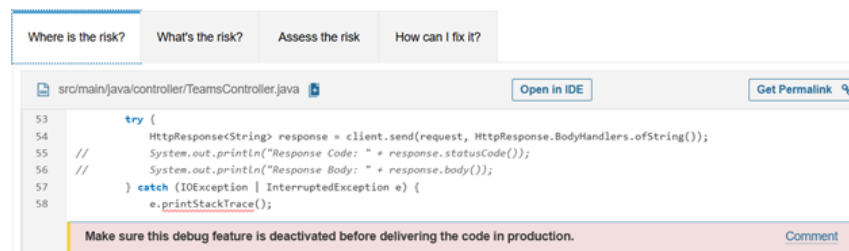


Figure 7: Do not enable debug features on production servers or application distributed to end users. Loggers should be used instead of *printStackTrace*

- Declaring a variable only to immediately return:

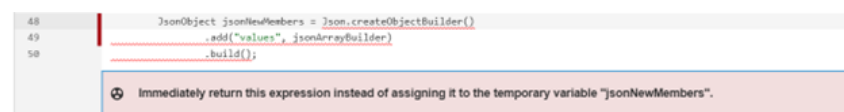


Figure 8: Declaring a variable only to immediately return or throw it is a bad practice. The method name should be sufficient for callers to know exactly what will be returned.

- Logger:



Figure 9: Logging a message helps user easily retrieve and read the logs. Sensitive data must only be logged securely.



Figure 10: Replace this use of System.out or System.err by a logger.



- Multiple catch blocks with same code:

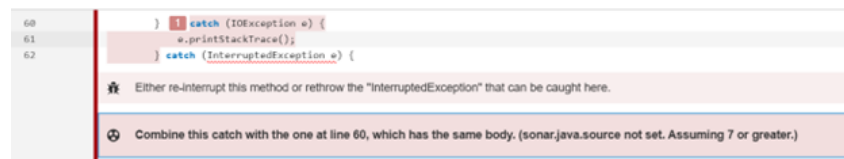


Figure 11: When multiple catch blocks have the same code, they should be combined for better readability.

- Nested try block:

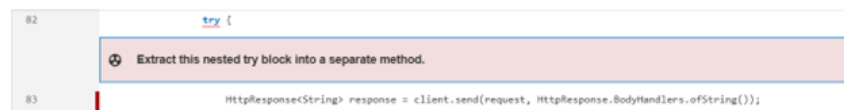


Figure 12: Nesting try/catch blocks severely impacts the readability of source code.

- Unintentional omission:

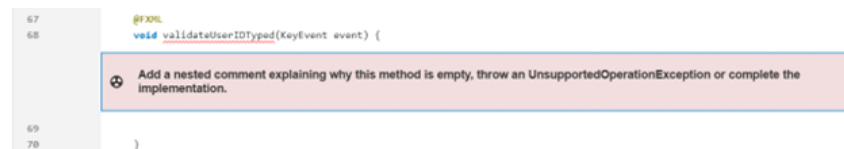


Figure 13: Unintentional omission should be fixed to prevent an unexpected behavior in production.

- Unused local variable:



Figure 14: Unused local variable should be removed.

## 3 Product Demo

### 3.1 Verify permission

 A screenshot of a web application window titled "Microsoft Team Management". The header is pink with the text "Microsoft Team Management" in black. Below the header is a logo for "MSTEAM management" featuring a pink star. The main content area has a light gray background. It contains three labels on the left: "TenantID", "ClientID", and "SecretID". To the right of "TenantID" is a text input field containing the value "9d6023b4-f4ce-4d66-9690-1a8c1f4f7da6". To the right of "ClientID" is an empty text input field. Below the "ClientID" input field, there is a red error message: "Please enter your client ID.". To the right of "SecretID" is an empty text input field. At the bottom center, there is a pink button with the text "Verify" in white.

Figure 15: Verify the owner's identity

The Microsoft Team Management app is exclusively designed for owners. Users must log in using their tenantID, clientID, and secreteID provided by the community to verify their identity as owners. Upon entering the correct IDs and clicking the verify button, the home page screen will appear. Otherwise, the app will provide feedback indicating incorrect IDs.

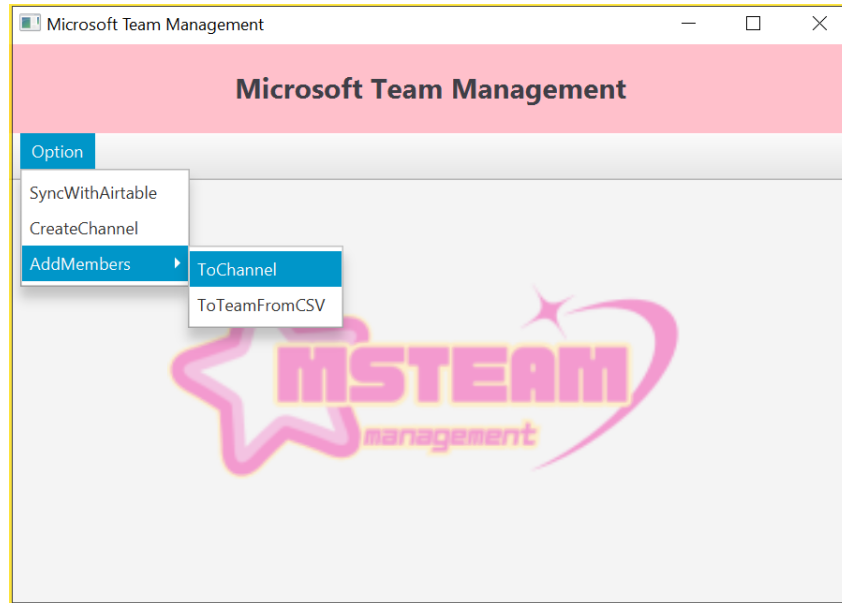


Figure 16: Homepage

The home page will display an "Option" menu that offers various functionalities for owners. Clicking the "Option" menu reveals a dropdown with three primary choices:

- **SyncWithAirtable:** Allows owners to synchronize data about members, channels, and teams with Airtable.
- **CreateChannel:** Provides functionality for creating new channels within Teams.
- **AddMembers:** Enables the addition of members to a specific channel or team.

### 3.2 Sync data to your Airtable base

Figure 17: Sync data to Airtable

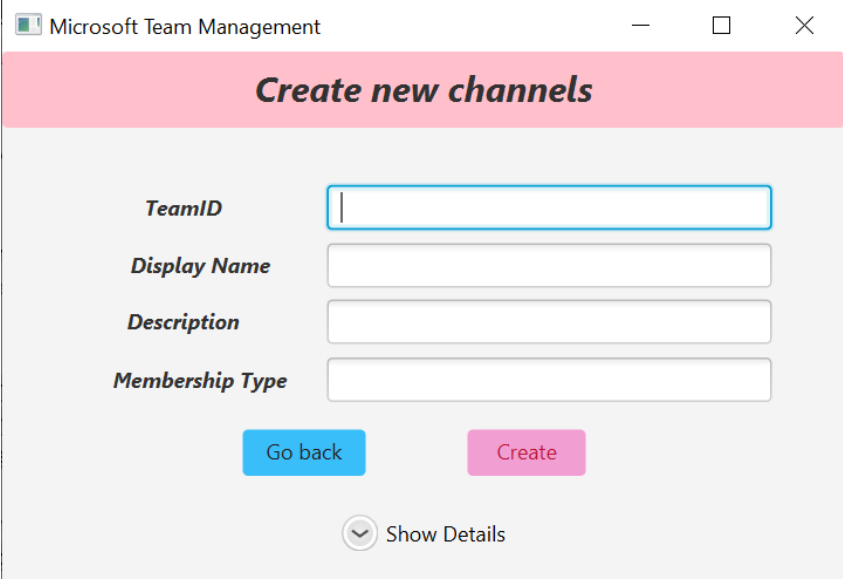
The first feature is SyncWithAirtable. To synchronize data about members, channels, and Teams with Airtable, owners need to enter specific information:

- **TeamID**: Allows owner access to Team to be synchronized.
- **baseID** and **AirtableKey**: Accesses to Airtable.
- **memberTableID** and **ChannelTableID**: Enables owners to access tables for storing data about members and channels.
- **period**: Specifies the update interval for syncing data.

The syncWithAirtable includes 3 button:

- **Sync**: Initiates the synchronization process.
- **Terminate**: Ends the synchronization process.

### 3.3 Create a channel



The screenshot shows a web application window titled "Microsoft Team Management". Inside, there is a pink header bar with the text "Create new channels". Below the header, there are four input fields with labels: "TeamID", "Display Name", "Description", and "Membership Type". At the bottom of the form, there are two buttons: "Go back" (blue) and "Create" (pink). Below the buttons, there is a "Show Details" link with a downward arrow icon.

Figure 18: Create new channels

The CreateChannel feature allows owners to create a new channel with a preferred display name, description, and membership type (such as standard, private, or shared) using the TeamID of the team where they want to create the channel.

### 3.4 Add new members to channel

To add new members to a channel, owners must input the TeamID to access the favorite team, the ChannelID for the specific channel they want to add members to, and finally, the userID of the new member.

**Microsoft Team Management**

**Add Members To Channel**

**TeamID**

**ChannelID**

**userID**

**Go Back** **Submit**

**Show Details**

Figure 19: Add new members to channel

### 3.5 Add new members to Team from CSV files

**Microsoft Team Management**

**TeamID**  **Back**

**Choose files**  **Add**

displayName	userId	email	roles
user1	5c214f19-530c-4c90-91...	user1@meadowilla.onmic...	member
Ha Phan	2a2b3132-efcc-4bf3-a17...	haphan@meadowilla.on...	member

**Show Details**

Figure 20: Add new members to Team

The final feature, `AddNewMembersToTeamFromCSV`, requires owners to enter the `TeamID` to access the desired team for adding new members. The app also supports a "Choose files" button to select a CSV file from their system. After clicking the "Add" button, the information from the CSV file displayed on the screen will be added to the team.

Each feature screen will have a "Back" button for easy navigation back to the home page. Additionally, upon completing an action or addition, a corresponding dialog will be shown like this:

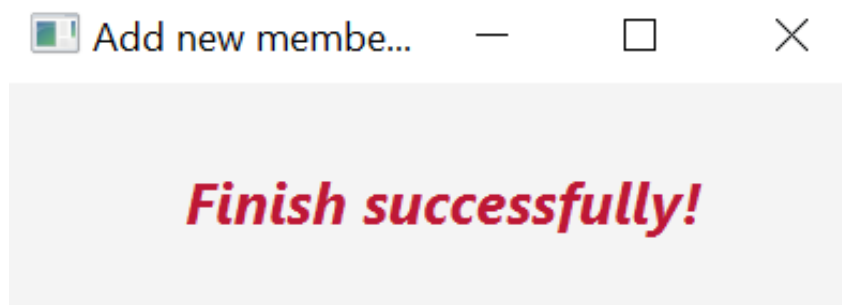


Figure 21: Finish the task

## 4 Future Development Discussion

### 4.1 More user-friendly GUI

During the usage of the app, we identified several areas for potential improvement:

- **User Guidance:** There is a need for more instructional messages to guide users on what information should be entered into each text field. For instance, it should be clear what "teamId" refers to and how users can obtain this information.
- **Error Handling and Notification:** Some issues are not sufficiently communicated to users, such as the missing "teamId" when attempting to add multiple members from a CSV file. Users should receive clear and actionable error messages to understand and resolve these issues effectively.

Acknowledging these weaknesses in our user interface, we recognize that with additional time, we could update our GUI code to enhance the overall user experience.

### 4.2 Smarter Syncing

Our syncing service now uses Airtable API to create new records, but not to delete all records. This indicates that if we click "Sync" and wait in a specific time, what we achieve on Airtable is the list with some repeated records. Therefore, it is not convenient for users while managing.

According to Airtable API Document [11], we can only delete at most 10 records at once API request. This is the disadvantage, and difficulty for us to apply deleting more than 10 records for syncing service.

## 5 Conclusion

In today's dynamic business landscape, the need for efficient team management and seamless communication has never been greater. Our comprehensive Microsoft Team Management app addresses these needs by providing robust solutions to common challenges faced by large organizations using Microsoft Teams.

By ensuring proper permissions, synchronizing data with Airtable, simplifying bulk member additions from CSV files, facilitating member management in private channels, and streamlining channel creation, our app significantly enhances the overall user experience and operational efficiency.

The features and functionalities detailed in this report illustrate how our app transforms Microsoft Teams management, making it more secure, efficient, and user-friendly. As organizations continue to rely on collaborative platforms like Microsoft Teams, our app stands out as an essential tool for maintaining effective team management and communication.

We are committed to continuous improvement and welcome feedback to further refine our app. Future updates will focus on addressing identified weaknesses and incorporating user suggestions to ensure our app remains at the forefront of team management solutions.

In summary, our Microsoft Team Management app is a vital asset for any organization looking to optimize their Microsoft Teams environment, fostering better collaboration and achieving greater organizational success.

# Bibliography

- [1] Microsoft Teams Collaboration <https://www.microsoft.com/insidetrack/blog/microsoft-teams-increases-collaboration-in-the-modern-workplace-at-microsoft/>
- [2] Microsoft Graph API <https://learn.microsoft.com/en-us/graph/api/resources/teams-api-overview?view=graph-rest-1.0>
- [3] Create a new channel <https://learn.microsoft.com/en-us/graph/api/channel-post?view=graph-rest-1.0&tabs=http>
- [4] Add member to channel <https://learn.microsoft.com/en-us/graph/api/channel-post-members?view=graph-rest-1.0&tabs=http>
- [5] Add member to team <https://learn.microsoft.com/en-us/graph/api/team-post-members?view=graph-rest-1.0&tabs=http>
- [6] Lists members of team <https://learn.microsoft.com/en-us/graph/api/team-list-members?view=graph-rest-1.0&tabs=http>
- [7] Get access without a user <https://learn.microsoft.com/en-us/graph/auth-v2-service?tabs=http>
- [8] Wikipedia, Airtable <https://en.wikipedia.org/wiki/Airtable>
- [9] Wikipedia, Jenkins [https://en.wikipedia.org/wiki/Jenkins\\_\(software\)](https://en.wikipedia.org/wiki/Jenkins_(software))
- [10] Wikipedia, SonarQube <https://en.wikipedia.org/wiki/SonarQube>
- [11] Delete multiple records <https://airtable.com/developers/web/api/delete-multiple-records>